We claim:

1. A scalable method for implementing FFT/IFFT computations in multiprocessor architectures that provides improved throughput by eliminating the need for inter-processor communication after the computation of the first "$\log_2 P$" stages for an implementation using "P" processing elements, comprising the steps of:

computing each butterfly of the first "$\log_2 P$" stages on either a single processor or each of the "P" processors simultaneously,

distributing the computation of the butterflies in all the subsequent stages among the "P" processors such that each chain of cascaded butterflies consisting of those butterflies that have inputs and outputs connected together, are processed by the same processor.

2. A method as claimed in claim 1 wherein the distributing of the computation of the butterflies subsequent to the first "$\log_2 P$" butterflies is achieved by assigning operand addresses of each set of butterfly operands to each processor in such a manner that the butterfly is processed by the same processor that computed the connected butterfly of the previous stage in the same chain of butterflies.

3. A method as claimed in claim 2 wherein the desired assignment of operand addresses is achieved by deriving the address of the first operand in the operand pair corresponding to the "i^{th}" stage of the computation from the address of the corresponding operand in the previous stage by inserting a "0" in the "$(i+1)^{th}$" bit position of the address, while the address of the second operand is derived by inserting a "1" in the "$(i+1)^{th}$" bit position of the operand address.

4. A method as claimed in claim 1 further including the computing of twiddle factors for the butterfly computations at each processor by initializing a counter and then incrementing it by a value corresponding to the number of processors "P" and appending the result with a specified number of "0"s.

5. A system for obtaining scalable implementation of FFT/IFFT computations in multiprocessor architectures that provides improved throughput by eliminating the need for inter-processor communication after the computation of the first "$\log_2 P$" stages for an implementation using "P" processing elements, comprising :

a means for computing each butterfly of the first "$\log_2 P$" stages on either a single processor or each of the "P" processors simultaneously,

an addressing means for distributing the computation of the butterflies in all the subsequent stages among the "P" processors such that each chain of cascaded butterflies consisting of those butterflies that have inputs and outputs connected together, are processed by the same processor.

6. A system as claimed in claim 5 wherein the addressing means comprises addresses generation means for deriving the operand addresses of the butterflies subsequent to the first "$\log_2 P$" butterflies in such a manner that the butterfly is processed by the same processor that computed the connected butterfly of the previous stage in the same chain of butterflies.

7. A system as claimed in claim 6 wherein the address generation means is a computing mechanism for deriving the address of the first operand in the operand pair corresponding to the "i^{th}" stage of the computation from the address of the corresponding operand in the previous stage by inserting a "0" in the "$(i+1)^{th}$" bit position of the address, and deriving the address of the second operand by inserting a "1" in the "$(i+1)^{th}$" bit position of the operand address.

8. A system as claimed in claim 5 further including a computing mechanism for address generation of twiddle factors for each butterfly on the corresponding processor.

9. A method of performing a fast Fourier transform or inverse fast Fourier transform on a plurality of inputs to generate a plurality of outputs, the method being performed on a plurality of processors and each transform including a

plurality of stages containing at least one butterfly computational block, the method comprising:

calculating the butterfly computational blocks for the first $\log_2(P)$ stages of the transform on a single one of the processors or on a plurality of the processors operating in parallel; and

calculating chains of butterfly computational blocks corresponding to the subsequent stages of the transform within each of the processors, each chain of butterfly computational blocks that is calculated in a respective processor having inputs and outputs coupled in series.

10. The method of claim 9 wherein the first $\log_2(P)$ stages of the transform are calculated on all of the processors operating in parallel.

11. The method of claim 9 wherein the method is performed on two processors, and wherein the first two stages of a radix-2 fast Fourier transform or inverse fast Fourier transform are calculated as a single radix-4 stage, and wherein the subsequent stages of the transform are computed as radix-2 stages.

12. The method of claim 11 wherein chains comprises a single loop that iterates $N/2 *(\log_2(N/2))/$(number of processors) times.

13. The method of claim 12 wherein each butterfly computational block includes a plurality of operands each having an associated address, and wherein calculating chains of butterfly computational blocks corresponding to the subsequent stages comprises assigning addresses to each of the operands so that each butterfly block in a chain is calculated in the same processor.

14. The method of claim 13 wherein each butterfly computational block includes a pair of operands, and wherein the operand addresses of these operands are assigned by deriving the address of the first operand in the operand pair

corresponding to the "i^{th}" stage of the calculation in the chain from the address of the corresponding operand in the previous stage by inserting a "0" in the "$(i+1)^{th}$" bit position of the operand address, and deriving the operand address of the second operand by inserting a "1" in the "$(i+1)^{th}$" bit position of the operand address.

15. The method of claim of claim 9 further comprising initializing a counter and then incrementing the counter by a value corresponding to the number of processors and appending the result with a specified number of "0"s to compute the twiddle factors for each butterfly computational block.

16. A processor system, comprising:

a plurality of processors operable to execute a fast Fourier transform or inverse fast Fourier transform algorithm on a plurality of inputs to generate a plurality of outputs, each transform including a plurality of stages containing at least one butterfly computational block, and the processors operable to the butterfly computational block for the first "$\log_2 P$" stages of the transform on either a single one of the processors or on a plurality of the processors operating in parallel; and

address circuitry operable to distribute the computation of the butterfly computational blocks in all stages subsequent to the first $\log_2 P$ states among the plurality of processors such that each chain of cascaded butterfly computational blocks in the transform are coupled in series and are computed by the same processor.

17. The processor system of claim 16 wherein the address circuitry is further operable to derive operand addresses for each of the butterfly blocks subsequent to the first "$\log_2 P$" butterfly blocks so that each of the butterfly computational blocks is computed by the same processor that computed a butterfly computational block of the previous stage in the same chain of butterfly computational blocks.

18. The processor system of claim 17 wherein each butterfly computational block includes a pair of operands, and wherein the address circuitry assigns operand addresses of these operands by deriving the address of the first operand in the operand pair corresponding to the "i^{th}" stage of the calculation in the chain from the address of the corresponding operand in the previous stage by inserting a "0" in the "$(i+1)^{th}$" bit position of the operand address, and deriving the operand address of the second operand by inserting a "1" in the "$(i+1)^{th}$" bit position of the operand address.

19. The processor system of claim 17 wherein the processors further comprise a counter that is initialized and then incremented by a value corresponding to the number of processors, an output of the counter being appended with a specified number of "0"s to compute twiddle factors for each butterfly computational block.

20. The processor system of claim 16 wherein each of the processors comprises a digital signal processor.

21. An electronic system, comprising:

a processor system, including,

a plurality of processors operable to execute a fast Fourier transform or inverse fast Fourier transform algorithm on a plurality of inputs to generate a plurality of outputs, each transform including a plurality of stages containing at least one butterfly computational block, and the processors operable to the butterfly

computational block for the first "$\log_2 P$" stages of the transform on either a single one of the processors or on a plurality of the processors operating in parallel; and

address circuitry operable to distribute the computation of the butterfly computational blocks in all stages subsequent to the first $\log_2 P$ states among the plurality of processors such that each chain of cascaded butterfly computational blocks in the transform are coupled in series and are computed by the same processor.

22. The electronic system of claim 22 wherein the electronic system comprises a communications system.

23. The electronic system of claim 20 wherein each of the processors comprises a digital signal processor.